Amendments to the Claims:

The listing of claims will replace all prior versions, and listings, of claims in the application.

Listing of Claims:

1. (Currently Amended) A method of computing comprising:

reading, by an execution engine, a data processing representation having code sections with code statements of at least a first and a second programming language;

recognizing, by the execution engine, a first code section with at least code statements of a first programming language;

invoking, by the execution engine, a first code statement processing unit of the first programming language to process the first code section;

invoking, by the execution engine, a second code statement processing unit of a second programming language to process a code statement, when the first code statement processing unit locates a code statement of the second programming language within the first code section, and invokes the execution engine recursively;

recognizing, by the execution engine, a second code section with at least code statements of the second programming language;

invoking, by the execution engine, the second code statement processing unit of the second programming language to process the second code section; and

invoking, by the execution engine, the first code statement processing unit of the first programming language to process a code statement, when the second code statement processing unit locates a code statement of the first programming language within the second code section, and invokes the execution engine recursively;

wherein the code statement of the second programming language within the first code section, and the code statement of the first programming language within the second code section, are both within the data processing representation.

2. (Previously Presented) The method of claim 1, wherein the first and second code sections are non-interleaved code sections.

3. (Original) The method of claim 1, wherein said second code section is embedded within said first code section.

4. (Previously Presented) The method of claim 1, wherein said first language is a directive language, and said second language is a selected one of XML and an object-oriented language.

5. (Previously Presented) The method of claim 1, wherein said first language is an object-oriented language, and said second language is XML.

6. (Previously Presented) The method of claim 1, wherein the method further comprises

recognizing a third code section with at least code statements of a third programming language; and

invoking a third code statement processing unit of the third programming language to process the third code section.

7. (Original) The method of claim 6, wherein said third code section is embedded within said second code section, and said second code section is embedded within said first code section.

8. (Previously Presented) The method of claim 6, wherein said first language is a directive language, said second language is an object-oriented language, and said third language is XML.

9. (Previously Presented) The method of claim 1, wherein the method further comprises

recognizing an invocation of a library function within at least a selected one of said first and second code sections; and

invoking the library function, and outputting the result of the invocation.

10. (Original) The method of claim 9, wherein the library function is a selected one of an emit function for outputting execution results, a pop function for returning an element, and a push function for backing up an insertion point.

11. (Original) The method of claim 1, wherein the method further comprises

recognizing a header section of a selected one of the first and the second programming language;

recognizing a directive statement within the header section, enumerating one or more data packages; and

importing the enumerated one or more data packages for use within code sections with at least statements of the selected first and second programming language.

12. (Original) The method of claim 1, wherein the method further comprises

recognizing a header section of a selected one of the first and the second programming language;

recognizing a declare statement within the header section, enumerating one or more processing methods; and

instantiating the enumerated one or more processing methods for use within code sections with at least statements of the selected first and second programming language.

13. (Original) The method of claim 1, wherein the method further comprises

recognizing a header section of a selected one of the first and the second programming language;

recognizing a declare statement within the header section, enumerating one or more instance variables; and

instantiating the enumerated one or more instance variables for use within code sections with at least statements of the selected first and second programming language.

14.-19. (Cancelled)

20. (Currently Amended) An apparatus comprising:

at least one storage unit having stored thereon programming instructions designed to instantiate an execution engine to enable the apparatus to

read, by the execution engine, a data processing representation having code sections with code statements of at least a first and a second programming language,

recognize, by the execution engine, a first code section with code statements of at least the first programming language,

invoke, by the execution engine, a first code statement processing unit of the first programming language to process the first code section,

invoke, by the execution engine, a second code statement processing unit of a second programming language to process a code statement, when the first code statement processing unit locates a code statement of the second programming language within the first code section, and invokes the execution engine recursively;

recognize, by the execution engine, a second code section with code statements of at least the second programming language,

invoke, by the execution engine, a second code statement processing unit of the second programming language to process the second code section,

invoke, by the execution engine, the first code statement processing unit of the first programming language to process a code statement, when the second code statement processing unit locates a code statement of the first programming language within the second code section, and invokes the execution engine recursively;

wherein the code statement of the second programming language within the first code section, and the code statement of the first programming language within the second code section, are both within the data processing representation; and

at least one processor coupled to said at least one storage unit to execute said programming instructions.

21. (Previously Presented) The apparatus of claim 20, wherein the first and second code sections are non-interleaved code sections.

22. (Original) The apparatus of claim 20, wherein said second code section is embedded within said first code section.

23. (Previously Presented) The apparatus of claim 20, wherein said first language is a directive language, and said second language is a selected one of XML and an object-oriented language.

24. (Previously Presented) The apparatus of claim 20, wherein said first language is an object-oriented language, and said second language is XML.

25. (Previously Presented) The apparatus of claim 20, wherein the programming instructions further enable the apparatus to

recognize a third code section with at least code statements of a third programming language; and

invoke a third code statement processing unit of the third programming language to process the third code section.

26. (Original) The apparatus of claim 25, wherein said third code section is embedded within said second code section, and said second code section is embedded within said first code section.

27. (Previously Presented) The apparatus of claim 25, wherein said first language is a directive language, said second language is an object-oriented language and said third language is XML.

28. (Previously Presented) The apparatus of claim 20, wherein said programming instructions further enable the apparatus to

recognize an invocation of a library function of a selected one of the first and the second programming language within the first code section; and

invoke the library function, and output the result of the invocation.

29. (Original) The apparatus of claim 28, wherein the library function is a selected one of an emit function for outputting execution results, a pop function for returning an element, and a push function for backing up an insertion point.

30. (Original) The apparatus of claim 20, wherein the said programming instructions are further designed to enable the apparatus to

recognize a header section of a selected one of the first and the second programming language;

recognize a directive statement within the header section, enumerating one or more data packages; and

import the enumerated one or more data packages for use by code sections with at least code statements of the selected one of the first and the second programming language.

31. (Original) The apparatus of claim 20, wherein said programming instructions are further designed to enable the apparatus to

recognize a header section of a selected one of the first and the second programming language;

recognize a declare statement within the header section, enumerating one or more processing methods; and

instantiate the enumerated one or more processing methods for use within code sections with at least code statements of the selected one of the first and the second programming language.

32. (Original) The apparatus of claim 20, wherein said programming instructions are further designed to enable the apparatus to

recognize a header section of a selected one of the first and the second programming language;

recognize a declare statement within the header section, enumerating one or more instance variables; and

instantiate the enumerated one or more instance variables for use code sections with at least code statements of the selected one of the first and the second programming language.

33.-38. (Cancelled)

39. (Currently Amended) A non-transitory computer-readable medium having instructions stored thereon that, when executed by a processor, cause the processor to implement an execution engine, the instructions comprising:

reading a data processing representation having code sections with code statements of at least a first and a second programming language;

recognizing a first code section with at least code statements of a first programming language;

invoking a first code statement processing unit of the first programming language to process the first code section;

invoking a second code statement processing unit of a second programming language to process a code statement, when the first code statement processing unit locates a code statement of the second programming language within the first code section, and invokes the execution engine recursively;

recognizing a second code section with at least code statements of the second programming language;

invoking the second code statement processing unit of the second programming language to process the second code section; and

invoking the first code statement processing unit of the first programming language to process a code statement, when the second code statement processing unit locates a code statement of the first programming language within the second code section, and invokes the execution engine recursively;

wherein the code statement of the second programming language within the first code section, and the code statement of the first programming language within the second code section, are both within the data processing representation.

40. (Currently Amended) The non-transitory computer-readable medium of claim 39, wherein the first and second code sections are non-interleaved code sections.

41. (Currently Amended) The non-transitory computer-readable medium of claim 39, wherein said second code section is embedded within said first code section.

42. (Currently Amended) The non-transitory computer-readable medium of claim 39, wherein said first language is a directive language, and said second language is a selected one of XML and an object-oriented language.

43. (Currently Amended) The non-transitory computer-readable medium of claim 39, wherein said first language is an object-oriented language, and said second language is XML.

44. (Currently Amended) The non-transitory computer-readable medium of claim 39, wherein the ~~method~~ instructions further ~~comprises~~comprise

recognizing a third code section with at least code statements of a third programming language; and

invoking a third code statement processing unit of the third programming language to process the third code section.

45. (Currently Amended) The non-transitory computer-readable medium of claim 44, wherein said third code section is embedded within said second code section, and said second code section is embedded within said first code section.

46. (Currently Amended) The non-transitory computer-readable medium of claim 44, wherein said first language is a directive language, said second language is an object-oriented language, and said third language is XML.

47. (Currently Amended) The non-transitory computer-readable medium of claim 39, wherein the instructions further comprise

recognizing an invocation of a library function within at least a selected one of said first and second code sections; and

invoking the library function, and outputting the result of the invocation.

48. (Currently Amended) The non-transitory computer-readable medium of claim 39, wherein the instructions further comprise

recognizing a header section of a selected one of the first and the second programming language;

recognizing a directive statement within the header section, enumerating one or more data packages; and

importing the enumerated one or more data packages for use within code sections with at least statements of the selected first and second programming language.

49. (Currently Amended) The non-transitory computer-readable medium of claim 39, wherein the instructions further comprise

recognizing a header section of a selected one of the first and the second programming language;

recognizing a declare statement within the header section, enumerating one or more processing methods; and

instantiating the enumerated one or more processing methods for use within code sections with at least statements of the selected first and second programming language.

50. (Currently Amended) The non-transitory computer-readable medium of claim 39, wherein the instructions further comprise

recognizing a header section of a selected one of the first and the second programming language;

recognizing a declare statement within the header section, enumerating one or more instance variables; and

instantiating the enumerated one or more instance variables for use within code sections with at least statements of the selected first and second programming language.